In [1]:
```
import pandas as pd
df=pd.read_csv("diabetes.csv")
```

In [4]:
```
df.shape
```

Out[4]:
```
(768, 3)
```

In [3]:
```
df.isnull().sum()
```

Out[3]:
```
Glucose    0
BMI        0
Outcome    0
dtype: int64
```

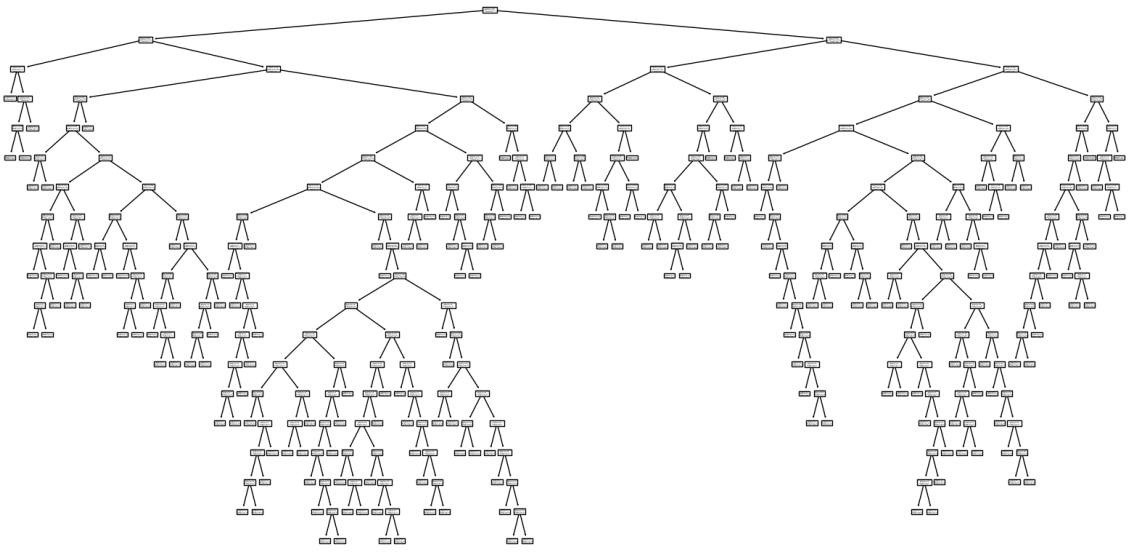
In [5]:
```
X=df.iloc[:,:-1].to_numpy()
y=df.iloc[:,-1].to_numpy()
```

In [6]:
```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,random_sta
```

In [12]:
```
from sklearn.tree import DecisionTreeClassifier
clf=DecisionTreeClassifier(random_state=0)
clf.fit(X_train,y_train)
```

Out[12]:
```
DecisionTreeClassifier
DecisionTreeClassifier(random_state=0)
```

In [13]:
```
import matplotlib.pyplot as plt
%matplotlib inline
from sklearn.tree import plot_tree
plt.figure(figsize=(20,10))
plot_tree(clf,feature_names=['Glucose','BMI'],class_names=['No','Yes'])
plt.show()
```

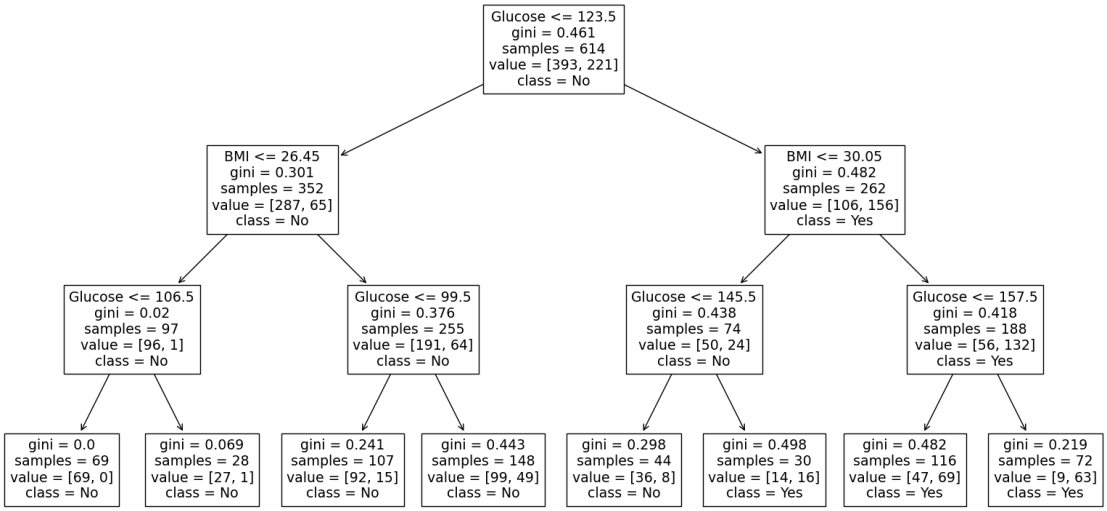


In [15]:
```
clf.set_params(max_depth=3)
```

Out[15]:
```
DecisionTreeClassifier

DecisionTreeClassifier(max_depth=3, random_state=0)
```

In [16]:
```
clf.fit(X_train,y_train)
plt.figure(figsize=(20,10))
plot_tree(clf,feature_names=['Glucose','BMI'],class_names=['No','Yes'])
plt.show()
```



In [17]:
```
predictions=clf.predict(X_test)
```

In [18]:
```
clf.predict([[90,20],[200,30]])
```

Out[18]:
```
array([0, 1], dtype=int64)
```

In [19]:
```
from sklearn.model_selection import cross_val_score
scores=cross_val_score(clf,X_train,y_train,cv=5,scoring='accuracy')
accuracy=scores.mean()
accuracy
```

Out[19]:
```
0.7182993469278955
```

In [20]:
```
from sklearn import metrics
cf=metrics.confusion_matrix(y_test,predictions)
cf
```

Out[20]:
```
array([[90, 17],
       [20, 27]], dtype=int64)
```

In [22]:
```
tp=cf[1][1]
tn=cf[0][0]
fp=cf[0][1]
fn=cf[1][0]
print(f"tp:{tp}, tn:{tn},fp:{fp},fn:{fn}")
```

```
tp:27, tn:90,fp:17,fn:20
```

In [24]:
```
print("accuracy",metrics.accuracy_score(y_test,predictions))
```

```
accuracy 0.7597402597402597
```

In [25]:
```
print("Precision",metrics.precision_score(y_test,predictions))
```

```
Precision 0.6136363636363636
```

In [27]:
```
print("Recall",metrics.recall_score(y_test,predictions))
```

```
Recall 0.574468085106383
```

In [29]:
```
feature_importances = clf.feature_importances_
print("Feature importances:",feature_importances)
```

```
Feature importances: [0.74096359 0.25903641]
```

In [ ]:
```

```